Boosting and Instability for Regression Trees

Servane Gey, Jean-Michel Poggi

Boosting and Instability for Regression Trees

Servane Gey [a] Jean-Michel Poggi [b]

[a] *Laboratoire MAP5 - Université Paris V, 75270 Paris Cedex 06, France. Servane.Gey@math-info.univ-paris5.fr*

[b] *Laboratoire de mathématiques - U.M.R. 8628 Université Paris XI, 91405 Orsay, France. Jean-Michel.Poggi@math.u-psud.fr*

Abstract

The AdaBoost like algorithm for boosting CART regression trees is considered. The boosting predictors sequence is analyzed on various data sets and the behaviour of the algorithm is investigated. An instability index of a given estimation method with respect to some training sample is defined. Based on the bagging algorithm, this instability index is then extended to quantify the additional instability provided by the boosting process with respect to the bagging one. Finally, the ability of boosting to track outliers and to concentrate on hard observations is used to explore a non-standard regression context.

Key words: Bagging; Boosting; CART; Instability; Prediction; Regression.

1 Introduction

Coming from the machine learning community, many authors have proposed boosting algorithms during the nineties. In the first half decade, the main concern is around the so-called PAC (Perturb And Combine) algorithm, see [1] for a concise overview. Starting from a training set $\{(X_i, Y_i)\}_{1 \leqslant i \leqslant n}$, the problem addressed is to fit a model delivering a prediction $\hat{y} = \hat{f}(x)$ at input x. The basic idea is to generate many different base predictors obtained by perturbing the training set and to combine them. Breiman [2] introduces bagging, theoretically analysed by Bühlmann and Yu [3], which averages predictions given by a sequence of predictors built from bootstrap samples of the training set, and then proposes different variants (see Breiman [4,5]).

After the paper of Freund and Schapire [6] introducing the AdaBoost algorithm for classification problems, a considerable interest for boosting schemes grew not only in artificial intelligence but also in statistics. The basic idea of

boosting is to improve the performance of a given estimation method using some bagging like scheme except that each predictor copes with a bootstrap sample obtained from the original one by adaptive resampling, highlighting the observations poorly predicted. A lot of contributions try to elucidate the surprisingly good behaviour of this algorithm in the classification context, see Schapire et al. [7], Friedman et al. [8], Lugosi and Vayatis [9] and Blanchard et al. [10]. An idea of the discussions five years ago can be found in the important discussion paper written by Breiman [11] about arcing classifiers.

This paper deals with boosting for regression problems. Roughly speaking, two different approaches have been considered. The first one is related to the gradient-based algorithm following the ideas initiated by Friedman (see Friedman [12], Zemel and Pitassi [13] and Rätsch et al. [14]). Following an entirely different line, other authors more closely relate the second one to the original algorithm of Freund and Schapire. On one hand, Ridgeway et al. [15] consider the regression problem as a large classification one. On the other hand, Drucker [16] proposes a direct adaptation of AdaBoost to the regression framework, which exhibits interesting performance by boosting CART regression trees (see also Borra and Di Ciacco [17]). These two papers conclude with interesting remarks about global performance comparisons but do not analyze the boosting predictors sequence and do not address the stability issue. These two points are the main purpose of this paper.

The paper is organized as follows. Section 2 sets the model and presents the regression tree estimation method. The Drucker's boosting algorithm is recalled in Section 3. The six reference data sets are described in Section 4. Section 5 analyzes the boosting predictors sequence. Section 6 provides a definition of two instability indices, for a given estimation method with respect to some training sample: an instability index based on bagging and an incremental instability index measuring the contribution to instability provided by boosting with respect to bagging. The ability of boosting techniques to track outliers and to concentrate on hard observations is then used in Section 7 to explore a non-standard regression context. Some concluding remarks are collected in Section 8.

2 Model and estimation method

We consider the following regression model:

$$Y = f(X) + \xi, \tag{1}$$

where $(X, Y) \in \mathbb{X} \times \mathbb{R}$, f is the unknown regression function to be recovered and ξ is an unobservable additive noise centered conditionally to X with un-

known variance σ_ξ^2. The generalization error of an estimator $\hat{f}$ of f is defined as $R(f,\hat{f}) = \mathbb{E}\left[\|f-\hat{f}\|^2\right]$.

Let us consider a training (or learning) sample L and a test sample T, of respective size n and n_t, composed of independent realizations of the variable (X,Y). The goal is then to construct from L an estimator $\hat{f}$ of f having low generalization error. Since the joint distribution is unknown, we evaluate the training error (often called the resubstitution error) and the prediction error of $\hat{f}$, respectively defined as $R_L(f,\hat{f})$ and $R_T(f,\hat{f})$, where for a given sample S of size m of the random variable (X,Y)

$$R_S(f,\hat{f}) = \frac{1}{m}\sum_{(X_i,Y_i)\in S}\left(Y_i - \hat{f}(X_i)\right)^2.$$

Since $\hat{f}$ is constructed from L and since T and L are independent, $R_T(f,\hat{f})$ is used as an estimator of $R(f,\hat{f})$.

In this paper we focus on CART regression trees to generate predictors. One of the particularly attractive property of this estimation method is, in this context, its instability. Therefore, the booststrap regression trees do not have the same number of terminal nodes and involve different features of the data.

Let us denote by $\mathcal{A}$ the base algorithm providing some base estimator (or predictor) $\hat{f}$ of f, which is here the well-known CART for regression (see Breiman *et al.* [18]), where the choice of the final tree is performed using a tuning set randomly taken from L, keeping T to give independent estimates of $R(f,\hat{f})$.

3 Algorithms

To improve the performance of CART, we focus on two algorithms: bagging and boosting.

Bagging is based on a bootstrap scheme: first, generate K replicate samples L_k from the uniform distribution, then construct predictors $\hat{f}_k^{(bag)}$ using $\mathcal{A}$ on L_k, and finally aggregate by averaging

$$\tilde{f}^{(bag)} = \frac{1}{K}\sum_{k=1}^{K}\hat{f}_k^{(bag)}.$$

It is now well known that bagging improves, often a lot, the performance of the considered method $\mathcal{A}$ (see Breiman [2,5] and Drucker [16] for regression

trees), leading to take bagging as a reference to evaluate boosting.

The boosting algorithm used here is given in Table 1. This algorithm is that of Drucker [16], which was directly adapted from AdaBoost algorithm [6] for classification problems.

Table 1
Boosting algorithm.

Boosting

Input: L the training sample of size n, $\mathcal{A}$ the estimation method and K the number of iterations,

Initialization: Set $p_1 = D$ the uniform distribution on $\{1, \ldots, n\}$

Loop: for $k = 1$ to K do

- randomly draw from L with replacement, according to p_k, a sample L_k of size n,

- using $\mathcal{A}$, construct an estimator $\hat{f}_k$ of f from L_k,

- set from the original training sample L: $i = 1, \ldots, n$

$l_k(i) = \left(Y_i - \hat{f}_k(X_i)\right)^2$ and $\varepsilon_{p_k}^{(boost)} = \sum_{i=1}^{n} p_k(i) l_k(i)$,

$\beta_k = \frac{\varepsilon_{p_k}^{(boost)}}{\max\limits_{1 \leqslant i \leqslant n} l_k(i) - \varepsilon_{p_k}^{(boost)}}$ and $d_k(i) = \frac{l_k(i)}{\max\limits_{1 \leqslant i \leqslant n} l_k(i)}$,

if $\varepsilon_{p_k}^{(boost)} < 0.5 \max_{1 \leqslant i \leqslant n} l_k(i)$, $w_{k+1}(i) = \beta_k^{1-d_k(i)} p_k(i)$,

else $w_{k+1} = p_1$,

$p_{k+1}(i) = \frac{w_{k+1}(i)}{\sum_{j=1}^{n} w_{k+1}(j)}$.

Output: $\tilde{f}$ the median of $\left(\hat{f}_k\right)_{1 \leqslant k \leqslant K}$ weighted by $\left(\log\left(\frac{1}{\beta_k}\right)\right)_{1 \leqslant k \leqslant K}$.

This algorithm comes from two ideas, theoretically motivated for the classification case in [6]. The first is about the resampling distribution updating. At iteration k, the distribution p_{k+1} is computed to increase the probability of appearance in the next sample L_{k+1} of the observations of L that are poorly predicted by $\hat{f}_k$. So the next predictor $\hat{f}_{k+1}$ will concentrate more on these observations. Here β_k (see Table 1) can be viewed as a synthetic index of the performance of $\hat{f}_k$ on L_k, where, for brevity, we write the performance on L under p_k as the performance on L_k. The smaller β_k, the better $\hat{f}_k$ performs on L_k. Since $d_k(i)$ is the individual performance of $\hat{f}_k$ on L, multiplying $p_k(i)$ by $\beta_k^{1-d_k(i)}$ merges in p_{k+1} the global performance of $\hat{f}_k$ on L_k and also the individual ones on L. The second is about the aggregation part of the algorithm. It is important to notice that the predictor $\hat{f}_k$ is constructed under

the distribution p_k, so the combination of the $(\hat{f}_k)$ has to take into account the resampling distributions (p_k). Indeed, each predictor $\hat{f}_k$ is designed to fit observations of L_k obtained from p_k, and then if $\hat{f}_k$ performs well on L_k but not globally on L, its weight will be large anyway. Let us mention that if $\beta_k \geq 1$, then boosting is regenerated by resetting p_k to p_1, avoiding possible degeneration.

4 Data sets and global performance

4.1 Data sets

We consider three simulated data sets, denoted by FR#1, FR#2 and FR#3, and three real data sets, Boston Housing, Paris Pollution and Vitrolles Pollution. They are summarized in Tables 2 and 3 respectively. Data sets FR#1, FR#2, FR#3 and Boston Housing are classical test examples (see [2], [18], [16] and [4]) while P-Pollution and V-Pollution are French ozone data.

Table 2
Simulated data sets.

Data	Predictors	Model
FR#1	$X_i \sim \mathcal{U}([0,1])$	$f(x) = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2$
	$i = 1, \ldots, 10$	$+10x_4 + 5x_5 + 0\sum_{i=6}^{10} x_i$
		$\xi \sim \mathcal{N}(0,1)$
FR#2	$X_1 \sim \mathcal{U}([0,100])$	$f(x) = \sqrt{x_1^2 + (x_2 x_3 - 1/x_2 x_4)^2}$
	$X_2/2\pi \sim \mathcal{U}([20,280])$	$\xi \sim \mathcal{N}(0,\sigma_\xi)$ SNR=3
	$X_3 \sim \mathcal{U}([0,1])$	
	$X_4 \sim \mathcal{U}([1,11])$	
FR#3	Same as FR#2	$f(x) = \tan^{-1}\left[\frac{x_2 x_3 - (1/x_2 x_4)}{x_1}\right]$
		$\xi \sim \mathcal{N}(0,\sigma_\xi)$ SNR=3

The simulated data sets have been considered by Friedman [19]. They are relevant for our experiments for two reasons. First, they exhibit different difficulties with respect to CART regression trees. Indeed, model FR#1 is defined using a simple nonlinear function partially additive and separable but involves, in addition, five useless variables while models FR#2 and FR#3 correspond

to highly nonlinear functions with strong interactions (the more important for FR#2). Second, these simulated data sets allow us to consider three different situations (see section 4.2): FR#1 for which boosting outperforms bagging, FR#2 for which bagging outperforms boosting and FR#3 which is a borderline example.

In addition to data sets FR#2 and FR#3 corresponding to a signal-to-noise ratio (SNR) equal to 3, we consider data sets denoted by FR#2-b and FR#3-b for which the SNR is equal to 9. In the first case, the signal explains 75% of the variance while it explains 90% in the second one. These two variants have been considered separately in [2] and [16]. Thus we can compare FR#2 with FR#2-b and FR#3 with FR#3-b to get an idea of the impact of changing the SNR from a moderate to a high value.

Table 3
Real data sets.

Data	**Response**	**Predictors**	**Number of obs.**
Boston Housing (Bost. Hous.)	Median housing price in the tract.	13 predictors. Fully described in [18].	506
Paris Pollution (P-Pollution)	Daily maximum ozone concentration.	3 predictors. Fully described in [20].	1200
Vitrolles Pollution (V-Pollution)	Daily maximum ozone concentration.	41 predictors. Fully described in [21].	822

The first real data set, called Boston Housing, is fully described in [18, pp 217-220] and extensively used in regression literature, allowing fair comparisons.

Let us sketch the specific problem addressed in this paper for the P-Pollution data, dealing with the analysis and prediction of ozone concentration in Paris area. Highly polluted days are often hard to predict: usual estimation methods need to be suitably post-processed to improve the performance on these observations. We investigate in section 7.2 if, starting from a CART regression tree, boosting performs automatically this improvement. The last data set called V-Pollution differs from the previous one since the interesting days are sufficiently numerous and a large number of explanatory variables is considered.

To start with a reference, we evaluate the performance of three estimators: a single tree and those obtained by bagging and boosting. We perform $K = 200$ iterations for bagging and boosting. The performance is measured by the prediction error $R_T(f, \tilde{f})$ computed using a test sample T. This estimation is computed by Monte Carlo from 10 runs of each algorithm. For the simulated data, 500 training examples and 600 test examples are used. For the Boston Housing data, we randomly take 10% of the data as test examples. For the pollution data sets, we use a test sample containing 10% of the data and stratified with respect to the response to preserve its distribution in the test set since performance for polluted days is considered as crucial.

Remark 1. Usually (see [2], [16], [21]) the samples L and T (often of fixed sizes) are randomized, which is convenient when the objective is to estimate the true generalization errors. In this paper, since we focus on definitions of instability depending on the training sample, all the predictors are generated from fixed samples L and T, chosen in such a way that the prediction error of a single tree designed using L, evaluated using T, is the median of the errors obtained from numerous randomly generated candidates.

The performance of the three estimators on the data sets is given in Table 4.

Table 4
Single tree, bagging and boosting performance for data sets.

Data	Single Tree	Bagging	Boosting
FR#1	8.79	5.75	4.46
FR#2	65,981	51,734	57,038
FR#2-b	26,147	18,981	19,507
FR#3	0.067	0.046	0.050
FR#3-b	0.036	0.026	0.024
Bost. Hous.	13.8	11.9	10.6
P-Pollution	556	469	488
V-Pollution	835	415	394

Remark 2. Let us mention that the results for single tree and bagging are close to those obtained by Breiman [2] on common classical data sets and by Ghattas [21] on V-Pollution. The prediction errors obtained for boosting are of the same order of magnitude (larger for FR#1 and smaller for FR#2, FR#3 and Bost. Hous.) than those obtained by Drucker [16], who uses a different pruning step in CART.

First of all, bagging and boosting perform significantly better than a single tree. This a widely known behaviour. However, boosting does not always have a better performance than bagging since boosting outperforms bagging only for FR#1, FR#3-b, Boston Housing and V-Pollution.

For FR#2 and FR#3 it is clear that boosting performs worse than bagging while the degradation for P-Pollution is smaller. But when the SNR is increased (see rows labeled FR#2-b and FR#3-b), the performance of a single tree and the bagged predictor are dramatically improved and, what is more interesting, bagging and boosting become closer. The comparison is even inverted for FR#3-b.

5 Analysis of the boosting predictors sequence

In the classification case, one can observe (see [11]) from the boosting predictors sequence the five following facts. (1) Boosting outperforms bagging in all but a few data sets examined in the literature. (2) The sequence (β_k), where β_k is defined via the misclassification rate of $\hat{f}_k$, oscillates without any pointwise stabilization. (3) Training and prediction errors decrease rapidly. Furthermore, one observes that the training (or resubstitution) error rapidly reaches zero and that the prediction error keeps on decreasing even after the training error is equal to zero. (4) No characteristic behaviour indicates some possible overfitting. (5) Breiman notices a "strange signature" which can be interpreted as an expression of the stabilization of the learning method via boosting. More precisely, he plots the misclassification rate of each observation of L along the K iterations, defined by

$$r(i) = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}_{Y_i \neq \hat{f}_k(X_i)} \qquad i = 1, \ldots, n$$

with respect to the number of times that the observation i appears in the different bootstrap samples L_k. The intuitive idea is that the more the observation i is misclassified, the more it appears in the bootstrap sample. But r seems to stabilize around a constant value reached by the majority of the observations. Breiman calls it "the plateau". That is why boosting is often interpreted in the classification context as a misclassification rate equalizer.

We sketch in this section a similar analysis for boosting in regression, to study if the main attractive features of AdaBoost are preserved.

5.1 The weights (β_k) and the prediction errors

The behaviour of (β_k) (defined in Table 1) and the training and prediction errors with respect to k for FR#1 are plotted in Figure 1.

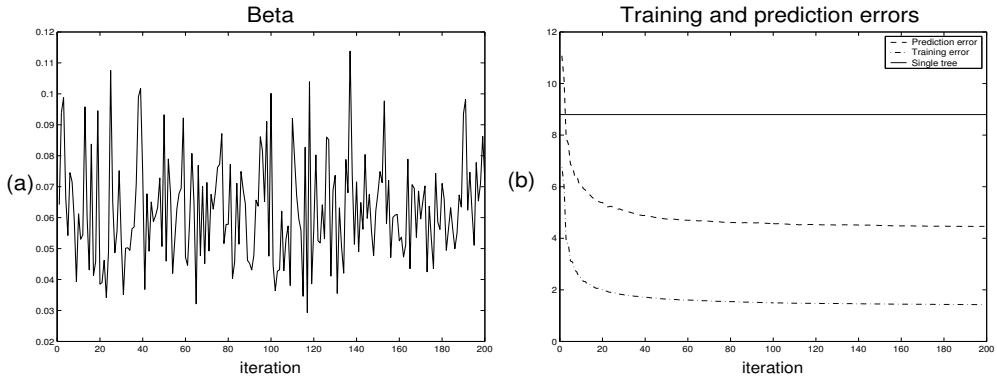

Fig. 1. FR#1: On the left, the (β_k) and on the right, from bottom to top, two curves representing the boosting training and prediction errors and a line indicating the prediction error of a single tree.

The sequence (β_k) (see Figure 1(a)) is very unstable and oscillates around 0.07. Since (β_k) is related to the global quality of $\hat{f}_k$ on L_k, it follows that this quality exhibits the same behaviour all along the boosting process without any pointwise stabilization. Similarly, $\hat{f}_k(x_i)$ typically oscillates around y_i without any pointwise stabilization. On Figure 1(b), the training and prediction errors have nearly the same behaviour as in the classification case, despite the fact that the prediction error does not obviously keep on decreasing after the training error stabilization. Moreover, it is interesting to notice that there is no characteristic behaviour indicating some possible overfitting, even after 200 iterations which is usually considered as a large value (the corresponding curves for the other data sets are similar).

5.2 The probability distributions (p_k)

Figure 2 shows the boosting resampling probabilities along the iterations, for two observations from FR#1 (panel (a)) and P-pollution (panel (b)). In each case the two observations have been chosen in such a way that the associated probabilities are respectively almost always above and below the line representing the uniform probability.

Again it appears that, in general, no pointwise stabilization occurs.

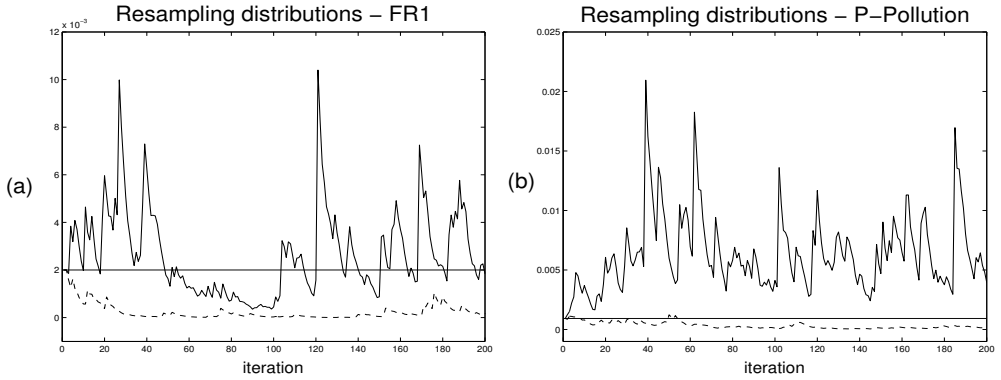

Fig. 2. For FR#1 (a) and P-pollution (b), the boosting resampling probabilities for two observations. The line represents the uniform probability.

5.3 The plateau

To study the plateau in regression, we define the individual average prediction error of Y_i using $(\hat{f}_k)$ along the K iterations by

$$r^{(boost)}(i) = \frac{1}{K} \sum_{k=1}^{K} \left(Y_i - \hat{f}_k(X_i) \right)^2 . \tag{2}$$

Then, we plot, for each observation i of L, $r^{(boost)}(i)$ with respect to the number of times that the observation i appears in the different bootstrap samples L_k (see Figure 3).

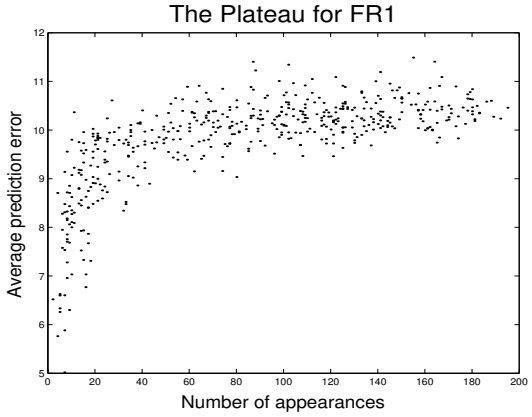

Fig. 3. FR#1: average error on 200 iterations vs. the number of appearances in the samples L_k.

Boosting behaves in regression as in classification since the plateau phenomenon occurs (the same behaviour is observed for the other data sets). So boosting can be considered as an error equalizer.

After this quick tour of the behaviour of the boosting predictors sequence, let

us focus on instability issue.

6 Bagging, boosting and instability

In the classification case, it is well known (see for example [11]) that, in order to make bagging and boosting effective, a key property of the given estimation method $\mathcal{A}$ is to be unstable. Intuitively $\mathcal{A}$ is an unstable method if small changes in the training sample L may cause large changes in the resulting estimator $\hat{f}$. In other words, $\mathcal{A}$ is unstable if it is not a robust method. Hence different ways to define the instability or the stability of $\mathcal{A}$ appear in the classification literature. On one hand, Bousquet and Elisseeff [22] study the stability of $\mathcal{A}$ by replacing one observation in L with another one coming from the same model. On the other hand, Breiman [23,11] studies a version of the variance of $\mathcal{A}$ and connects instability with the good performance of bagging. We follow the second line and we try to connect instability with bagging and boosting sequences of predictors.

6.1 Bagging and instability

We define an instability index depending on a specific choice of the training sample in order to be close to the actual problem of model estimation, for which only one realization is available.

6.1.1 Instability index

The key quantities measuring the instability can be derived from the errors, evaluated on L, of the bagging predictors $(\hat{f}_k^{(bag)})$:

$$\varepsilon_k^{(bag)} = R_L(f, \hat{f}_k^{(bag)}). \tag{3}$$

Indeed the fluctuations of these errors describe the changes of the estimator caused by changes in the training sample. So the coefficient of variation of $\varepsilon^{(bag)}$ provides an instability index defined as the ratio:

$$I_L = \frac{\mathrm{std}\left(\varepsilon^{(bag)}\right)}{\overline{\varepsilon^{(bag)}}} \tag{4}$$

where $\varepsilon^{(bag)}$ denotes the vector $(\varepsilon_k^{(bag)})_{1\leqslant k\leqslant K}$, $\overline{\varepsilon^{(bag)}}$ its average and std $\left(\varepsilon^{(bag)}\right)$ its empirical standard deviation.

It follows from this definition of instability that, for a given L, the more unstable $\mathcal{A}$, the larger I_L, and, as soon as $\mathcal{A}$ is a local estimation method, I_L is not sensitive to outliers, since bagged predictors are involved. Let us mention that, with respect to model (1), for a given training sample L and some nonparametric estimation method $\mathcal{A}$, I_L can be viewed as an estimate of the coefficient of variation of the random variable $\|f - \hat{f}\|^2$.

6.1.2 Instability for data sets

To relate instability with performance of bagging, as Breiman does in [11] to show that bagging improves only unstable methods, we compute, for each data set, the instability index and the percentage of improvement of a single tree by bagging, defined by $100(pe_{st} - pe_{bag})/pe_{st}$ where pe_{st} and pe_{bag} denote the prediction errors (given in Table 4) for a single tree and bagging respectively. CART is bagged on the training sample L and 10 runs of bagging with 200 iterations per run are performed to estimate I_L by the Monte Carlo method. The question is: can we consider that the larger I_L is, the more bagging will improve CART? The results are given in Table 5. The data sets are split in two groups numbered 1 when boosting performs better than bagging and numbered 2 otherwise.

Table 5
Instability index and CART improvement by bagging.

Data	**Group**	I_L	**%Improvement by bagging**
FR#1	1	0.075	34.6
FR#2	2	0.052	21.6
FR#2-b	2	0.055	27.4
FR#3	2	0.081	31.3
FR#3-b	1	0.093	27.8
Bost. Hous.	1	0.167	14.1
P-Pollution	2	0.039	12.0
V-Pollution	1	0.074	50.2

As expected, the percentage of improvement increases with I_L, except for Boston Housing which behaves as an outlier. For this last case, we observe the highest value of I_L and a percentage close to the smallest one. This is due to the intrinsic difficulty of this data set since a very large instability index

is combined with a large improvement of bagging by boosting (see Table 4). This point will be addressed in the next section.

Let us remark that data sets of *Group* 1 seem to have larger instability indexes than the ones of *Group* 2.

Two obvious remarks illustrate on data sets the good behaviour of the previously defined instability index: for a given regression function, I_L increases with the SNR (see FR#2 and FR#3 related rows) and I_L decreases with the number of missed variables (see Pollution data rows). This point is developed in [24] by considering nested models.

6.2 Boosting and instability

6.2.1 Incremental instability index

Let us now define some indexes measuring the contribution of boosting to instability. The general idea is that, compared to bagging, the adaptive updating of the probabilities (p_k) should provide a sequence of boosted predictors $(\hat{f}_k)$ much more unstable than the sequence of bagged predictors $(\hat{f}_k^{(bag)})$.

Starting from the key sequence of the training errors of the boosting predictors $(\hat{f}_k)$

$$\varepsilon_k^{(boost)} = R_L(f, \hat{f}_k), \tag{5}$$

the instability index $I_L^{(boost)}$ carried out by boosting for the sample L is obtained from I_L by taking $\varepsilon^{(boost)}$ rather than $\varepsilon^{(bag)}$, that is:

$$I_L^{(boost)} = \frac{\text{std}\left(\varepsilon^{(boost)}\right)}{\overline{\varepsilon^{(boost)}}}. \tag{6}$$

Then the incremental instability index of boosting for the sample L is defined as the ratio of the instability index carried out by boosting and the instability index:

$$\delta I_L^{(boost)} = \frac{I_L^{(boost)}}{I_L} = \frac{\text{std}\left(\varepsilon^{(boost)}\right)}{\overline{\varepsilon^{(boost)}}} \times \frac{\overline{\varepsilon^{(bag)}}}{\text{std}\left(\varepsilon^{(bag)}\right)}. \tag{7}$$

It follows that, for a given L, the larger $\delta I_L^{(boost)}$, the more boosting contributes to instability with respect to bagging.

6.2.2 Incremental instability for data sets

For each data set, the two numerical instability indices and the ratio of the prediction errors of bagging and boosting are given in Table 6.

Table 6
Instability indices for bagging and boosting.

Data	**Group**	I_L	$\delta I_L^{(boost)}$	**Ratio bagging/ boosting**
FR#1	1	0.075	1.87	1.29
FR#2	2	0.052	2.38	0.91
FR#2-b	2	0.055	2.58	0.97
FR#3	2	0.081	1.57	0.92
FR#3-b	1	0.093	1.58	1.08
Bost. Hous.	1	0.167	1.50	1.11
P-Pollution	2	0.039	3.50	0.97
V-Pollution	1	0.074	1.97	1.05

First of all, let us notice that the contribution of boosting to instability is effective for all data sets, since $\delta I_L^{(boost)}$ is always greater than 1.

Second the results of Table 6 have to be balanced with the classification of the data sets recalled in the second column of the table. We obtain that for *Group* 1 the instability index is large and the incremental instability index is moderate and, for *Group* 2, except for FR#3, the instability index is small and the incremental instability index is significantly larger than the ones of *Group* 1, particularly for P-Pollution.

This leads to the conjecture that the effectiveness of boosting is not only related to the global instability of CART, but also to the incremental contribution of boosting to instability: a large (resp. small) instability combined with a moderate (resp. large) contribution of boosting to instability may imply that boosting will perform well (resp. poorly).

The FR#3 data set is a borderline example since a change of the SNR suffices to invert the bagging/boosting ratio.

The P-Pollution data set illustrates that boosting focuses on hard observations, which are the highly polluted days. Indeed, the instability index is small

since it is robust with respect to them, but the incremental instability index is very large since it is sensitive to them. Table 4 shows that boosting performs slightly worse than bagging. A deeper analysis carried out in section 7.2 exhibits a more interesting behaviour of boosting: it performs well and improves bagging performance on highly polluted days even if it slightly degrades the global performance.

6.2.3 More about the training errors

Let us formulate a surprising remark about the averaged errors by considering the following ratio: $\Delta_{(bag)}^{(boost)} = \frac{\overline{\varepsilon^{(boost)}}}{\overline{\varepsilon^{(bag)}}}$. Intuitively, since bagging is not sensitive to outliers or hard observations, the more boosting concentrates on some observations of the training sample L, the more the average error of the sequence $(\hat{f}_k)$ is far from the average error of the sequence $(\hat{f}_k^{(bag)})$. So Table 7 contains $\Delta_{(bag)}^{(boost)}$ and the incremental instability index of boosting for all the considered data sets.

Table 7
Ratio of the average errors of the predictors sequences provided by boosting and bagging respectively and incremental instability index for the data sets.

Data	$\Delta_{(bag)}^{(boost)}$	$\delta I_L^{(boost)}$
FR#1	1.51	1.87
FR#2	1.69	2.38
FR#2-b	1.57	2.58
FR#3	1.55	1.57
FR#3-b	1.49	1.58
Bost. Hous.	1.50	1.50
P-Pollution	1.68	3.50
V-Pollution	1.55	1.97

Surprisingly $\Delta_{(bag)}^{(boost)}$ remains within the interval $[1.5; 1.7]$ for the considered data sets. This means that the Drucker's boosting seems to design predictors in order to equalize their average error to at least 1.5 times the average error of the bagging predictors sequence, as soon as K is sufficiently large. This remark shows how important is the aggregation part of the algorithm.

In addition, it follows that the incremental instability index given by (7) seems to be close to the ratio of the standard deviations of $\varepsilon^{(boost)}$ and $\varepsilon^{(bag)}$, up to a multiplicative constant roughly speaking.

Let us come back to the plateau evidenced in section 5.3 by connecting r

defined by (2) and ε, both for bagging and boosting. Indeed, we have that $\overline{r^{(boost)}} = \overline{\varepsilon^{(boost)}}$. Then by replacing $\hat{f}_k$ by $\hat{f}_k^{(bag)}$ in the definition of $r^{(boost)}(i)$, we define similarly $r^{(bag)}(i)$ and we have $\overline{r^{(bag)}} = \overline{\varepsilon^{(bag)}}$. So

$$\Delta_{(bag)}^{(boost)} = \frac{\overline{\varepsilon^{(boost)}}}{\overline{\varepsilon^{(bag)}}} = \frac{\overline{r^{(boost)}}}{\overline{r^{(bag)}}}.$$

So $\Delta_{(bag)}^{(boost)}$ is also equal to the ratio of $\overline{r^{(boost)}}$ and $\overline{r^{(bag)}}$, this approximately gives the ordinate of the plateau.

7 Outliers and hard observations

Let us analyze the boosting dynamic in front of hard observations, first by contaminating some population with outliers, and second by studying if boosting is able to track the non-dominant population and to perform a kind of compromise. We first consider some simulated data, to cope with outliers, and then the P-Pollution data, which can be seen as a more intricate case of mixing of populations.

7.1 Outliers

The main point of this section is to supply evidence that boosting does concentrate on the outliers. We start with the model generating the FR#1 data set and contaminate it incrementally. Hence, since boosting performs well on this problem, the behaviour of boosting in front of outliers can be easily analyzed. From model FR#1 used to generate the dominant population, we simulate a training sample L and a test sample T of size 500 and 600 respectively. Then L and T are altered by adding to each of them $\delta\%$ observations generated using a shifted version of the former regression function ($f_2 = f + 20$) to produce the outliers. In addition we simulate two test samples T_1 and T_2 of size 600, coming from the dominant population and the outliers respectively.

Let us summarize the comparison of CART, bagging and boosting with respect to prediction errors evaluated on T, T_1 and T_2, for δ=1%, 2%, 3%, 5%, 10%, 25% (see [24] for details). The prediction error of bagging is always smaller than the prediction error of a single tree. Indeed bagging is not sensitive to outliers and generally improves the performance of CART. For δ less than 10%, the prediction errors of boosting and bagging are close to each other but bagging outperforms boosting for the three test sets. This illustrates the sensitivity of boosting to outliers, since boosting has better performance than bagging on the unaltered data set FR#1. Moreover, when there is a large proportion

of outliers ($\delta = 25\%$), boosting performs even worse than a single tree on the dominant population. Nevertheless an improvement on T_2 is reached. This balance between the performance on the dominant population and a second one is the key point highlighted in the next section.

What is more interesting is to focus on outliers. Figure 4 contains the number of times that each observation is taken in the bootstrap sample L_k and the percentage of outliers taken in L_k at each iteration k. Figure 4(a) shows that the outliers (the last observations in the training sample) are taken in almost every bootstrap sample L_k. Figure 4(b) shows that outliers do not represent more than 55% of the observations in L_k. Let us mention that the weight (with respect to the resampling distributions) of the outliers varies between 40% and 50%. This shows that boosting not only concentrates on outliers but mixes observations from the two subpopulations. This behaviour is analogous to the behaviour of boosting in classification, for which the p_{k+1}-weight of the observations misclassified by $\hat{f}_k$ is equal to 50%.

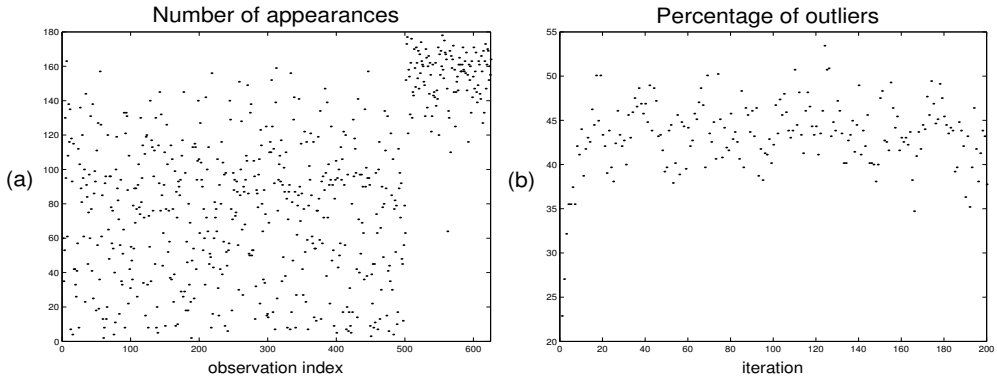

Fig. 4. (a): number of appearances of each observation in the bootstrap samples. (b): percentage of outliers taken in the bootstrap samples during the loop for FR#1 altered with $\delta = 25\%$ of outliers. *Note*: the outliers are the last observations in the training sample.

7.2 Hard observations

We now consider the P-Pollution data. A comparison between different approaches shows that a simple nonlinear additive model captures the main features of the complex underlying dynamics, despite the small number of explanatory variables (see [20]). The main difficulty is to correctly predict alarms defined by the maximum of ozone concentration exceedances of a high level threshold, for which only few observations are available. When a classical estimator is used, one can observe an underestimation for these interesting days despite the fact that the mean of absolute errors is satisfactory since it is close to the measurement error. To improve this estimator, Chèze et al. [20] define partial estimators for nonlinear additive models, which are recombined using

suitably chosen weights leading to a modified estimator improving tremendously the performance for interesting days. We study if boosting is able to automatically perform this. Since the legal threshold of maximum concentration of ozone over Paris is 130, population 1 (denoted by $P1$) is composed by observations having a response smaller than 130 and population 2 ($P2$) contains the other ones. From section 4.2, we know that boosting performs worse than bagging on this data set when performance is evaluated using T, which merges $P1$ and $P2$. Let us evaluate the performance separately on each population. The training and prediction errors obtained from 10 runs with 200 iterations are given in Table 8.

Table 8
Performance per population for P-Pollution data: training and prediction mean squared errors.

		Single tree	**Bagging**	**Boosting**
Training error	*P1*	305	249	207
	P2	864	640	236
Prediction error	*P1*	549	477	503
	P2	632	386	318

As claimed, a single tree performs better on the dominant population $P1$, both on training and test sets.

On the training set, bagging improves on a single tree both on $P1$ and $P2$. What is more interesting is that boosting improves the performance of bagging again on both populations and the accuracy on $P2$ becomes close to the one obtained for $P1$. This is illustrated in Figure 5 giving along the iterations the training errors for bagging (panel (a)) and boosting (panel (b)). Let us remark that the final values are already reached for $K = 80$ iterations.

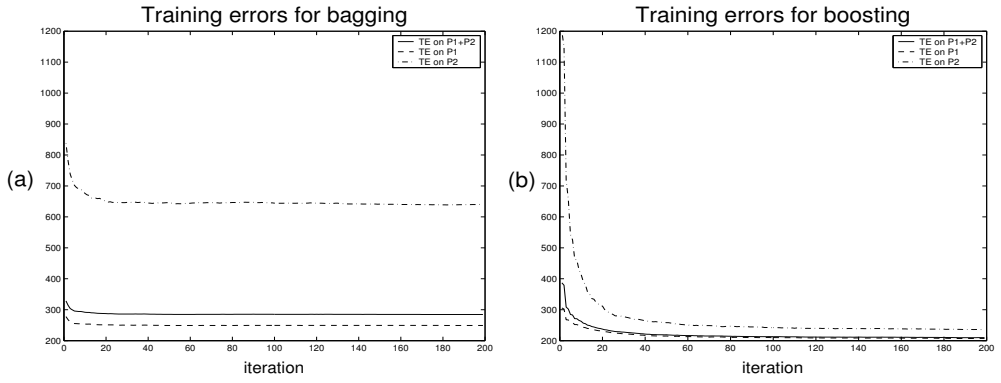

Fig. 5. P-Pollution: the training errors for bagging (a) and boosting (b). From bottom to top of each graph the curves correspond to $P1$, $P1 \cup P2$ and $P2$.

On the test set, a more surprising situation occurs: on one hand, bagging has better performance on $P2$ than on $P1$ and on the other hand, boosting slightly

degrades the performance of bagging on $P1$, but improves a lot the bagging performance on $P2$. The results on the test set must be taken with caution since only a small number of observations is involved.

From another viewpoint, it is clear from Figure 6 that boosting concentrates on the $P2$ observations.

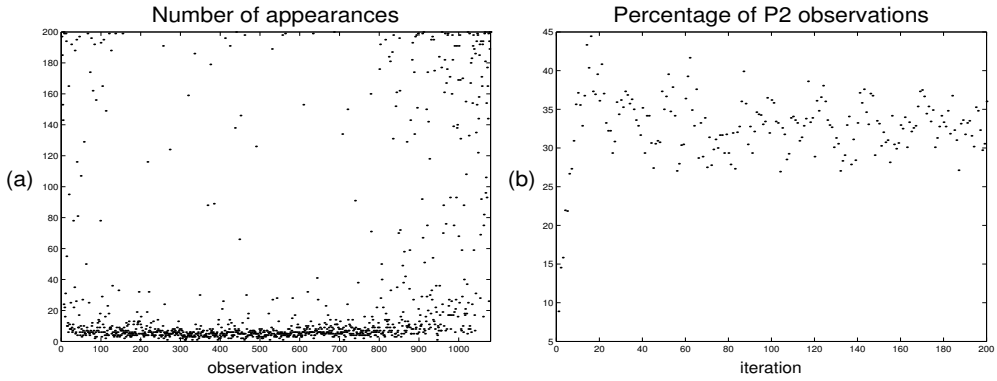

Fig. 6. Number of times that each observation is taken in the bootstrap samples (a) and percentage of $P2$ observations taken in the bootstrap samples during the loop (b) for P-Pollution data.

8 Concluding remarks

The analysis of Drucker's boosting algorithm focusing on predictors sequence reveals that many meaningful properties of the original AdaBoost algorithm are preserved. The definitions of instability introduced in this paper are also derived from the predictors sequence and confirm the importance of instability in order to get such resampling schemes effective. The instability indices allow a better understanding of the performance of bagging and boosting on data sets. For example, the effectiveness of boosting is not only related to the global instability of CART, but also to the incremental contribution of boosting to instability. The experiments carried out in this paper confirm the interest of the Drucker's algorithm to improve regression trees: the Paris pollution data illustrates the good behaviour of the algorithm in front of a non-standard regression model, performing a kind of tradeoff between the two underlying subpopulations. In addition, our experiments suggest that boosting could be used to highlight outliers or hard observations.

Finally, let us mention that, following numerous experiments (see [24]), it seems to be rather difficult to significantly improve the original algorithm using variants like aggregation using mean rather than median, aggregation using space adaptive definition of the weight or probabilities updating using classification like rule following [13].

Acknowledgments

We would like to thank Badih Ghattas for introducing us to boosting during exciting discussions and useful advices about computational issues. We are also grateful to Gilles Blanchard for helpful discussions. In addition, we thank an anonymous referee and an Associate Editor for remarks and suggestions which helped to improve the presentation of this article.

References

[1] R. E. Schapire, A brief introduction to boosting, in: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 1999.

[2] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.

[3] P. Bühlmann, B. Yu, Nonparametric classification – analyzing bagging, Annals of Statistics 30 (4) (2002) 927–961.

[4] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32.

[5] L. Breiman, Using iterated bagging to debias regressions, Machine Learning 45 (3) (2001) 261–277.

[6] Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences 55 (1) (1997) 119–139.

[7] R. E. Schapire, Y. Freund, P. Bartlett, W. Sun Lee, Boosting the margin : a new explanation for the effectiveness of voting methods, The Annals of Statistics 26 (5) (1998) 1651–1686.

[8] J. Friedman, T. Hastie, R. Tibshirani, Additive logistic regression : a statistical view of boosting, The Annals of Statistics 28 (2) (2000) 337–407.

[9] G. Lugosi, N. Vayatis, On the Bayes-risk consistency of regularized boosting methods, Ann. Statist. 32 (1) (2004) 30–55.

[10] G. Blanchard, G. Lugosi, N. Vayatis, On the rate of convergence of regularized boosting classifiers, Journal of Machine Learning Research (Special issue on learning theory) 4 (2003) 861–894.

[11] L. Breiman, Arcing classifiers (with discussion), The Annals of Statistics 26 (3) (1998) 801–849.

[12] J. Friedman, Greedy function approximation : the gradient boosting machine, The Annals of Statistics 29 (5) (2001) 1189–1232.

[13] R. S. Zemel, T. Pitassi, A gradient-based boosting algorithm for regression problems, in: Advances in Neural Information Processing Systems, no. 13 in NIPS-13, 2001, pp. 696–702.

[14] G. Rätsch, M. Warmuth, S. Mika, T. Onoda, S. Lemm, K. Müller, Barrier boosting, in: Proc. of the 13th Conf. on Comput. Learning Theory, 2000.

[15] G. Ridgeway, D. Madigan, T. Richardson, Boosting methodology for regression problems, in: Proc. of the 7th Int. Workshop on Artificial Intelligence and Statistics, 1999.

[16] H. Drucker, Improving regressors using boosting techniques, in: M. Kaufmann (Ed.), Proc. of the 14th Int. Conf. on Machine Learning, 1997, pp. 107–115.

[17] S. Borra, A. Di Ciacco, Improving nonparametric regression methods by bagging and boosting, Computational Statistics & Data Analysis 38 (2002) 407–420.

[18] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, Classification And Regression Trees, Chapman & Hall, 1984.

[19] J. Friedman, Multivariate adaptive regression splines (with discussion), The Annals of Statistics 19 (1991) 1–141.

[20] N. Chèze, J.-M. Poggi, B. Portier, Partial and recombined estimators for nonlinear additive models, Stat. Inf. for Stochastic Processes 6 (2) (2003) 155–197.

[21] B. Ghattas, Prévision des pics d'ozone par arbres de régression, simples et agrégés par bootstrap, Revue de Statistique Appliquée 47 (2) (1999) 61–80.

[22] O. Bousquet, A. Elisseeff, Stability and generalization, Journal of Machine Learning Research 2 (2002) 499–526.

[23] L. Breiman, Heuristics of instability and stabilization in model selection, Annals of Statistics 24 (6) (1996) 2350–2383.

[24] S. Gey, J.-M. Poggi, Boosting and instability for regression trees, Tech. Rep. 36, Université Paris XI (2002).